LA-UR-15-22523
Approved for public release; distribution is unlimited.

Title: MCNP 2020

Author(s): Brown, Forrest B.

Intended for: MCNP documentation & planning

Issued: 2015-04-08

MCNP 2020

An initiative to preserve 70 years of LANL investment in Monte Carlo radiation transport & prepare for future computer systems

Forrest B. Brown

R&D Scientist 5, Monte Carlo Codes Group, LANL

Fellow of the American Nuclear Society

Adjunct Professor – New Mexico, Michigan, RPI

2015-04-01

Monte Carlo & MCNP History

ENIAC – 1945

30 tons

20 ft x 40 ft room

18,000 vacuum tubes

0.1 MHz

20 word memory

patchcords

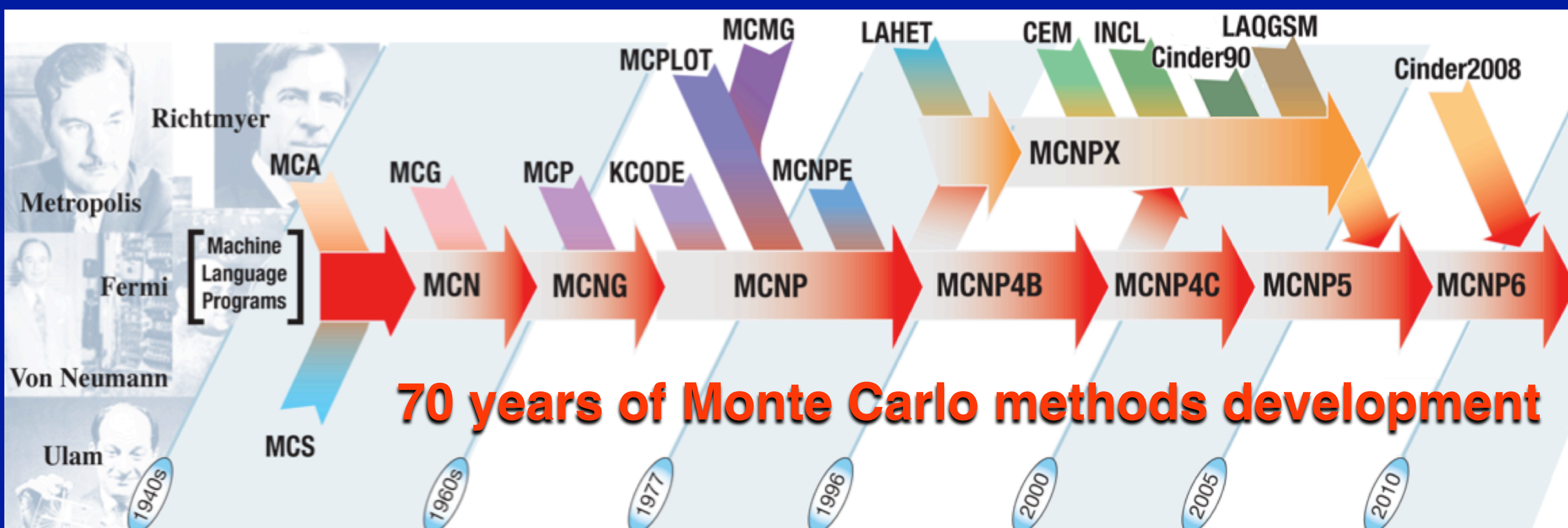
Manhattan Project – 1945...

Discussions on using ENIAC

Ulam suggested using the "method of statistical trials"

Metropolis suggested the name "Monte Carlo"

Von Neumann developed the first computer code



MCNP6 Features

mcnp5
neutrons, photons, electrons
cross-section library physics
criticality features
shielding, dose
“low energy” physics
V&V history
documentation

New Criticality Features
Sensitivity/Uncertainty Analysis
Fission Matrix
OTF Doppler Broadening

Fission
MCNP5/X multiplicity
LLNL fission package
CGM/LLNLGAM, CGMF (soon)

mcnp6

mcnp6
protons, proton radiography
high energy physics models
magnetic fields

Partisn mesh geometry
Abaqus unstructured mesh

mcnpx
33 other particle types
heavy ions
CINDER depletion/burnup
delayed particles

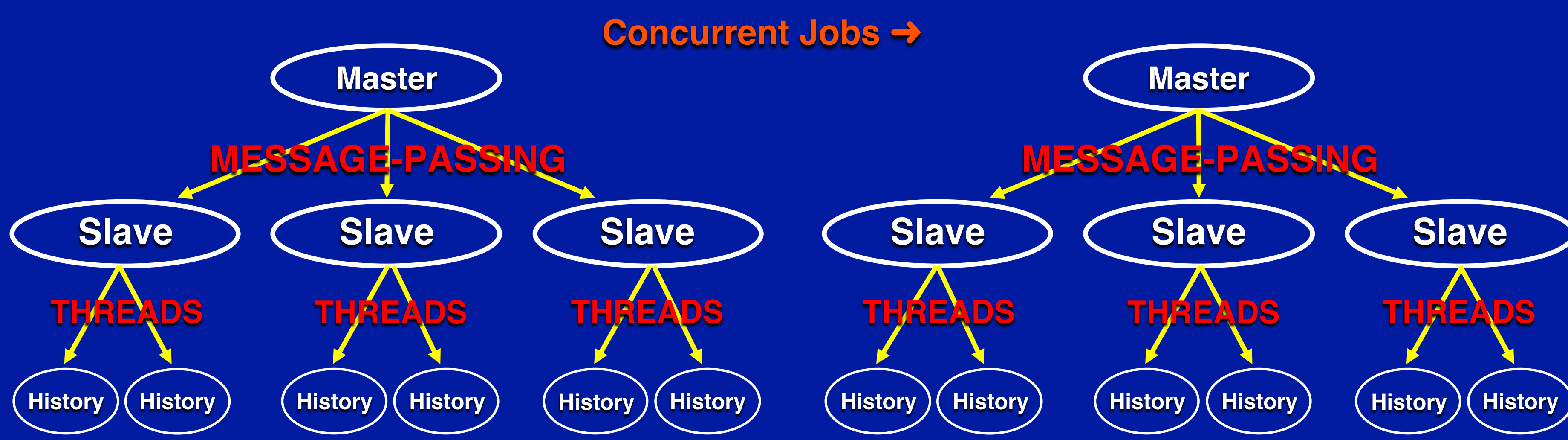
High energy physics models
CEM, LAQGSM, LAHET
MARS, HETC

Continuous Testing System
~10,000 test problems / day

mcnp6.1 – 2013
mcnp6.1.1 – 2014

mcnp5 – 100 K lines of code, 12,000+ copies distributed by RSICC
mcnp6 – 500 K lines of code, 7,819+ copies distributed by RSICC

MCNP – Hierarchical Parallelism – Since 2000



Parallel Processes

Total processes = (# jobs) x (# MPI processes) x (# threads)

Tradeoffs:

- More MPI processes - lots more memory & messages
- More threads - contention from lock/unlock shared memory
- More jobs - system complexity, combining results

MCNP6 Is At Risk …

MCNP6

- Product of 70 years of MC code development at LANL;
  respected for long history & extensive validation work
- Extensive use in: nuclear criticality safety, stockpile stewardship, homeland security, radiological safety, emergency response, experiment design, etc.

Risks:

- Leading-edge HPC systems change radically over 5-10 year periods.
  MCNP changes are needed for the coming exascale systems.
- Core portions of MCNP are largely based on computing practices from the 1950-60s, difficult to maintain and modify today.
- The merged code size grew to 500k lines. Greatly increased code complexity causes problems & delays for maintenance, improvements, and testing.
- Very fragile coding, Rube Goldberg logic
- “Feature bloat” and “technical debt”. Many 100s of features have been added, with very little effort toward upgrading the core portions of the code.
- After the merger, MCNP6 ran 30% - 500% slower than previous versions, making it difficult to convince users to switch to newer versions.

We need to address these risks now, & not wait for a crisis

MCNP 2020

- Improve performance - single thread & parallel
- Modernize coding & upgrade infrastructure
- Prepare for future computer systems

Evolution, not revolution

Leverage 70-year investment in MC

MCNP 2020

Path forward

- Concerted effort to modernize the codebase, upgrade foundations
- Address performance, including single-thread, threading, & MPI
- Necessary for MCNP to survive into the 2020's & new computers
- Need sustained, multi-year effort
- Need support from major stakeholders
- Proposed joint support by DOE-ASC & DOE-NCSP
  - Experienced Lead (Brown)
  - 2-3 core developers

Improve performance

Goal: 2X speedup within 2 years

Upgrade core MCNP6 software

- Restructure, clean up coding, Fortran 2003 & C/C++ standards
- Reorganize data structures
- Evolution, not revolution
- Reduce future costs for new development & maintenance

Goal: sustainable code

Prepare for future

- New computers – massive parallel, but less memory per core
- Improve MPI & thread parallelism

Goal: flexible, adaptable code

Nuclear Criticality Safety Program

Improve Performance (1)

Single thread "classic optimization" during history calculation, where appropriate:

- Inlining
- Replace accessor functions by direct lookups
- Guard-if-tests for global options, to avoid unnecessary work or function calls
- Possible changes to local algorithms, e.g., for sorting, ordering, checking (e.g., hash-based energy lookup scheme)
- Storage changes for better cache usage
- Eliminate all strided vector operations

Improve Performance (2)

Initial 3-month effort, focus on speedup & optimization

- Focus on neutron criticality problems
- **Speedup factors** from recent performance improvements

Performance Test Set
Speedups vs MCNP6.1 Release

Criticality		Other	
ks1	1.76	void1	3.03
ks2	2.13	void2	4.11
ks3	1.35	void3	2.72
ks4	1.36		
baw1	2.19	det1	1.67
baw2	1.59	med1	1.15
fvf	2.04	pht1	1.22
g1	1.14		
g2	2.20		
pin	1.73		

Performance Benchmark Suite
Speedups vs MCNP6.1 Release

Neutron Problems	
BAWXI2	4.37
GODIVA	1.05
Mode n in air w 750,000 tally bins	1.18
Well log problem	1.91
100M lattice cells in void	5.17
Other	
mode p e in air	1.01
mode n p e in air	1.05
mode p in air	1.20
Pulse height tally	1.20
Radiography	1.07

Improve Performance (3)

Run Times for VALIDATION_CRITICALITY Suite on Various Computers

Computer	CPU Speed (GHz)	Mem. Speed (GHz)	Processors, Cores	MCNP Threads Used	MCNP Version	Total Time (minutes)
MacBook 2010	2.7	1.1	1 - i7, 2 x 2 HT	4	mcnp6.1.1	88
MacBook 2013	3.0	1.6	1 - i7, 2 x 2 HT	4	mcnp5-1.60	40
				4	mcnp6.1	62
				4	mcnp6.1.1	42
Mac Pro 2010	3.0	0.67	2 - Xeon, 4	8	mcnp5-1.60	30
				8	mcnp6.1	44
				8	mcnp6.1.1	28
Windows 2012	2.7	1.3	2 - Xeon, 6	10	mcnp6.1.1	19
Mac Pro 2012	2.4	1.07	2 - Xeon, 4 x 2 HT	16	mcnp5-1.60	25
				16	mcnp6.1	32
				16	mcnp6.1.1	22
Mac Pro 2014	2.7	1.6	1 - Xeon, 12 x 2 HT	12	mcnp5-1.60	14
				12	mcnp6.1	20
				12	mcnp6.1.1	14
				14	mcnp6.1.1	12

Improve Performance (4)

Parallel algorithm & implementation modifications, where appropriate:

- "Lightweight" & "heavyweight" rendevous for KCODE, MPI for tallies at end
- Timing info based on wall-clock, not cpu
- MPI for groups of scalars, not individual scalars
- Asynchronous MPI messages
- Eliminate threading lock/unlock for sources
- Investigate/change threadprivate storage

Modernization & Infrastructure (1)

Standards, superset extensions, data structures, clarity, style, build system

- All code must adhere to Fortran 2003 & C++ standards
- Examine compliance with LANL P1020, DOE, ISO, NQA-1, ANS

- Introduce standard, consistent scheme for superset extensions

- Improve portability & robustness (uninit vars, FP ops, ifdefs,…)

- Rearrange & group module variables by function (not alphabet)

- Definition & description of code variables & logic

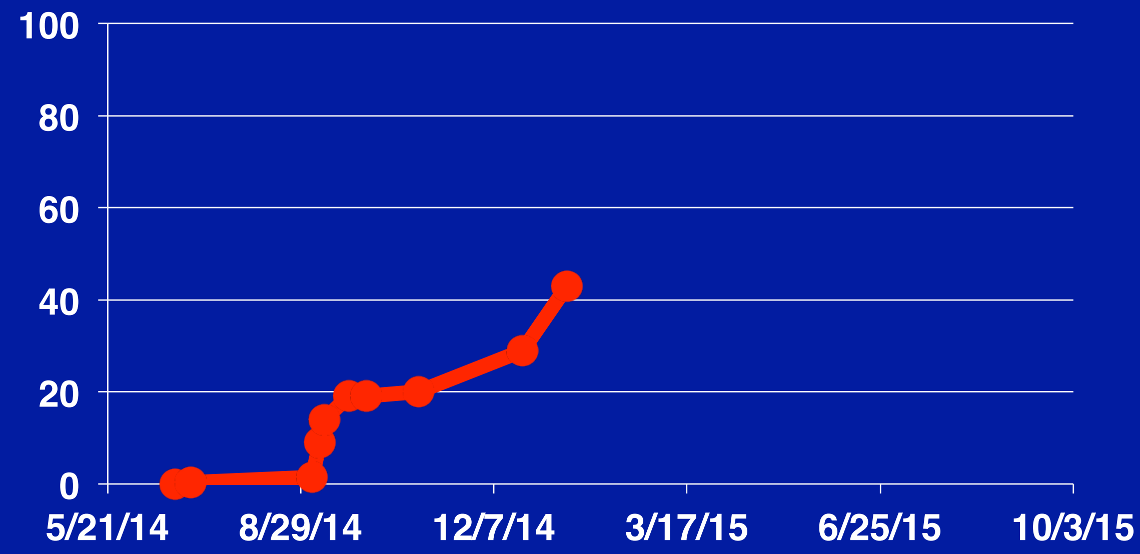
- Reduce verbosity, improve clarity of coding
- Revise parameter names for consistency & clarity

Modernization & Infrastructure (2)

Compliance with the Fortran 2003 Standard

- All code must adhere to Fortran 2003 & C++ standards
- MCNP6.1.1 had 3185 violations, spread over 243 files
- Target: fix all noncompliant coding in FY 2015

Progress



Will examine compliance with ANS, LANL P1020, DOE, NQA-1

Modernization & Infrastructure (3)

Standards, superset extensions, data structures, clarity, style, build system

- Simplify & clarify the build system
- Generate correct dependencies for mixed Fortran/C++ routines

- Fortran 2003 polymorphism for MPI routines
- Review/revise pblcom/tskcom usage

- Cache-friendly data structures for geometry, tallies, particles, physics

- Standard direct-access run-tape file, with dataset access by name
- Eliminate all use of scratch files & file I/O during histories

- Create MCNP_TEST directory, keep REGRESSION in source tree
- Rearrange directory structure under Source/

Modernization & Infrastructure (4)

Some proposals for reusable, portable software packages that could be shared among LANL Monte Carlo codes (mcnp, mcatk, others)

- **New Parallel Random Number Package**
- **Data Decomposition with Remote Tally Servers**
- **List Tally Package**
- **Standardized Tally Output Files**
- **Parallel PTRAC Capability**
- **Direct-access Dumpfiles**
- **Structured Cross-section Data Storage**
- **Unstructured Mesh Tracking Library**

Prepare for Future Computers (1)

Millions of threads, limited memory, limited I/O

Trinity Computer System at LANL

- **2015: 1st half, ~ 10k Xeon processors**
- **2016: 2nd half, ~ 10k Intel Phi MIC processors**

James Reinders, Intel:

> **"The MIC architecture…is our approach when we architect the chip assuming you are going to run a parallel program on it"**
>
> **"We've optimized it to run a parallel program as fast as possible, and it's absolutely terrible at running a non-parallel program."**

Prepare for Future Computers (2)

Millions of threads, limited memory, limited I/O

- Test & tweak for Intel MICs (240 threads/processor)
- Improve threading efficiency for 100s or 1000s of threads
- Explicit loops for threads
- Interrupts or errors kill thread, rerun as needed

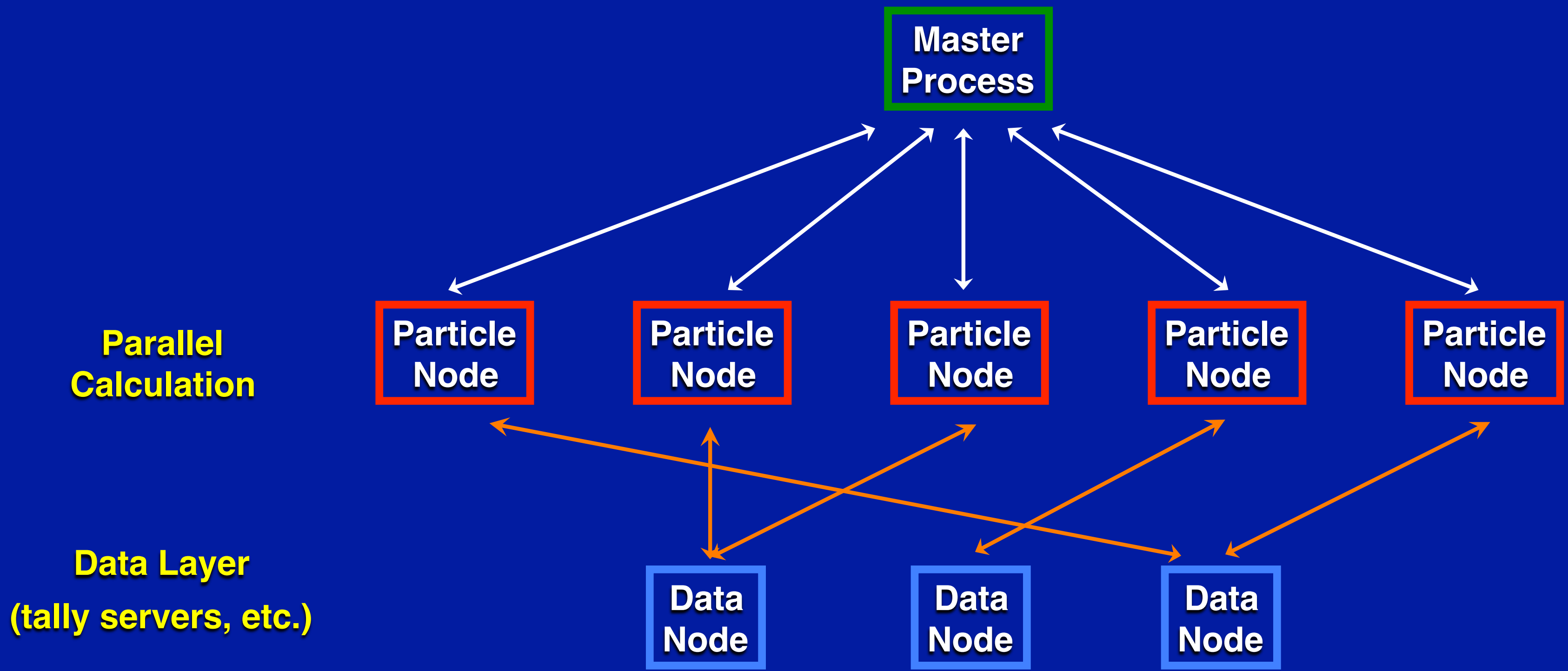
- Buffered list-tallies (regular & mesh)
- Optimized coding for tally-servers
- Data decomposition for tallies & meshes

- Extend hash-based energy lookups to other particles

- Eliminate all file I/O during histories, use buffered lists (esp. for SSR & PTRAC)

Prepare for Future Computers (3)

Particle parallelism + data decomposition -- logical view:



Mapping of logical processes onto compute nodes is flexible:

- Could map particle & data processes to different or same compute nodes
- Lightweight – particles, heavy-weight – data & tallies
- Heterogeneous nodes – range of memory, speed, parallelism, etc.

MCNP 2020 - Plans

General goals, strategy, plans

- MCNP-2020 white paper, 2013
- Presentations, NCSP work plans, etc.

Contributors

- Brown, Bull, Solomon, Rising, Martz

Funding

- Nuclear Criticality Safety Program (NCSP)
- Currently little or no ASC support

Priorities

- Critical code issues
- Discussions among contributors
- Thursday morning MCNP6 technical meetings

MCNP 2020 - Activities

NCSP - Related

- **Parallel threading** – measure 2015 performance of atomic-operations vs critical-sections
- **List tallies** – to save memory & reduce lock/unlock overhead for threading
- **Light-weight rendezvous** – reduce unnecessary MPI messaging
- **Compliance with Fortran-2003 standard** – eliminate all coding using older or nonstandard features
- **Fission neutron multiplicity** – restructure & combine, ensure correct threading

Non - NCSP (on hold)

- **MPI improvements** – nonblocking, asynchronous, in-place, interface using Fortran-2003 polymorphism
- **Memory allocation** – rearrange for better cache utilization
- **Tally servers** – remote node storage for tallies with very large memory
- **New dumpfile** – direct access, access by dataset name, etc.
- **PTRAC & SSR upgrade** – permit use in parallel calculations, not just serial
- **HDF5 &/or MPI-IO** – improve read/write speed & portability of file output

MCNP 2020 - Activities

Fortran 2003 compliance

- Rising, Bull, Martz, Brown

Performance

- Brown – parallel, classic optimization

Infrastructure & future

- Brown – threading, MPI, cache, algorithms
- Bull – list-tallies for mesh tallies
- Solomon – list-tallies for regular tallies
- Rising – consolidate & rearrange multiplicity treatments

Questions ?